

**AMENDMENTS TO THE CLAIMS**

The listing of claims will replace all prior versions, and listings, of claims in the application:

**Listing of Claims:**

1. (Currently Amended) A computer implemented method for adjusting a priority of an execution thread, said method comprising:

indicating that the execution thread needs a higher priority by updating a user mode accessible data area, the indicating performed without increasing a priority corresponding to the execution thread and the indicating including setting a critical section flag within the user mode accessible data  
indicating that the execution thread is entering a critical section of code;  
updating a priority offset amount prior to the execution thread entering the critical section of code, whereupon the priority offset amount is included in the user mode accessible data;

during execution of the critical section of code:

detecting a preemption event, wherein the preemption event has a preemption event priority;

reading the user mode accessible data area in response to the detected preemption event; [[and]]

determining whether a priority applied flag and the critical section flag have been set;

shifting in response to detecting that the critical section flag has been set and the priority applied flag has not been set, raising the execution thread's priority based upon the user mode accessible data- by the priority offset amount and setting the priority applied flag;

after raising the execution thread's priority:

in response to the preemption event priority being greater than the execution thread's priority, preempting the execution thread;  
and

in response to the preemption event priority not being greater than the execution thread's priority, allowing the execution thread to continue executing.

2. (Canceled)
3. (Canceled)
4. (Canceled)
5. (Currently Amended) The method of claim ~~[[4]]~~ 1 further comprising:  
~~setting~~ resetting the critical section flag within the user mode accessible data  
indicating that the execution thread is no longer in the critical section;  
receiving a second preemption event ~~at the kernel process;~~  
determining, ~~at the kernel process,~~ that the critical section flag is no longer set  
and that the priority applied flag ~~indicates that the execution thread's~~  
~~priority has been shifted~~ is set;  
~~re-shifting~~ lowering the execution thread's priority by the priority offset amount in  
response to the determination that the critical section flag is no longer set  
and that the priority flag ~~indicates that the execution thread's priority has~~  
~~been shifted~~ is set; and  
resetting the priority applied flag indicating that the execution thread's priority is  
no longer ~~shifted~~ raised.
6. (Canceled)
7. (Currently Amended) The method of claim 1 wherein the indicating that the execution thread needs a higher priority is performed in response to the execution thread entering ~~[[a]]~~ the critical code section, and wherein the critical code section ~~[[that]]~~ utilizes a shared system resource.
8. (Currently Amended) An information handling system comprising:  
one or more processors;

a memory accessible by the processors;

a nonvolatile storage device accessible by the processors;

a preemptive multitasking operating system controlling access by execution threads to the processors; and

a delayed priority tool for a delayed priority boost to the execution threads, the delayed priority tool comprising instructions stored in the memory, which, when executed by the processors, cause the information handling system to perform a method comprising:

~~means for~~ indicating that the an execution thread needs a higher priority by updating a user mode accessible data area, the indicating performed without increasing a priority corresponding to the execution thread and the indicating including setting a critical section flag within the user mode accessible data indicating that the execution thread is entering a critical section of code;

updating a priority offset amount prior to the execution thread entering the critical section of code, whereupon the priority offset amount is included in the user mode accessible data;

during execution of the critical section of code:

~~means for~~ detecting a preemption event, wherein the preemption event has a preemption event priority;

~~means for~~ reading the user mode accessible data area in response to the detected preemption event; [[and]]

determining whether a priority applied flag and the critical section flag have been set;

~~means for shifting~~ in response to detecting that the critical section flag has been set and the priority applied flag has not been set, raising the execution thread's priority based upon the user mode accessible data, by the priority offset amount and setting the priority applied flag;

after raising the execution thread's priority:

in response to the preemption event priority being greater than the execution thread's priority, preempting the execution thread; and  
in response to the preemption event priority not being greater than the execution thread's priority, allowing the execution thread to continue executing.

9. (Canceled)

10. (Canceled)

11. (Canceled)

12. (Currently Amended) The information handling system of claim [[11]] 8 further comprising:

~~means for setting~~ resetting the critical section flag within the user mode accessible data indicating that the execution thread is no longer in the critical section;

~~means for receiving a second preemption event at the kernel process;~~

~~means for determining, at the kernel process,~~ that the critical section flag is no longer set and that the priority applied flag ~~indicates that the execution thread's priority has been shifted~~ is set;

~~means for re-shifting~~ lowering the execution thread's priority by the priority offset amount in response to the determination that the critical section flag is no longer set and that the priority flag ~~indicates that the execution thread's priority has been shifted~~ is set; and

~~means for resetting the priority applied flag indicating that the execution thread's priority is no longer shifted~~ raised.

13. (Canceled)

14. A computer program product stored in a computer operable media, the computer operable media containing instructions for execution by a computer, which, when

executed by the computer, cause the computer to perform a method for adjusting a priority of an execution thread, said ~~computer program product~~ method comprising:

~~means for~~ means for indicating that the execution thread needs a higher priority by updating a user mode accessible data area, the indicating performed without increasing a priority corresponding to the execution thread and the indicating including setting a critical section flag within the user mode accessible data indicating that the execution thread is entering a critical section of code;

updating a priority offset amount prior to the execution thread entering the critical section of code, whereupon the priority offset amount is included in the user mode accessible data;

during execution of the critical section of code:

~~means for~~ detecting a preemption event, wherein the preemption event has a preemption event priority;

~~means for~~ reading the user mode accessible data area in response to the detected preemption event; [[and]]

determining whether a priority applied flag and the critical section flag have been set;

~~means for shifting~~ in response to detecting that the critical section flag has been set and the priority applied flag has not been set, raising the execution thread's priority based upon the ~~user mode accessible data~~ by the priority offset amount and setting the priority applied flag;

after raising the execution thread's priority:

in response to the preemption event priority being greater than the execution thread's priority, preempting the execution thread;  
and

in response to the preemption event priority not being greater than the execution thread's priority, allowing the execution thread to continue executing.

15. (Canceled)
16. (Canceled)
17. (Canceled)
18. (Currently Amended) The computer program product of claim 17 ~~further comprising~~ wherein the method further comprises:
  - ~~means for setting~~ resetting the critical section flag within the user mode accessible data indicating that the execution thread is no longer in the critical section;
  - ~~means for receiving a second preemption event at the kernel process;~~
  - ~~means for determining, at the kernel process,~~ that the critical section flag is no longer set and that the priority applied flag ~~indicates that the execution thread's priority has been shifted~~ is set;
  - ~~means for re-shifting~~ lowering the execution thread's priority by the priority offset amount in response to the determination that the critical section flag is no longer set and that the priority flag ~~indicates that the execution thread's priority has been shifted~~ is set; and
  - ~~means for resetting the priority applied flag indicating that the execution thread's priority is no longer shifted~~ raised.
19. (Canceled)
20. (Currently Amended) The computer program product of claim 14 wherein the ~~means for~~ indicating that the execution thread needs a higher priority is performed in response to the execution thread entering [[a]] the critical code section, and wherein the critical code section ~~[[that]]~~ utilizes a shared system resource.